Schedule Request dan Event Reminder Pada Aplikasi Finding Tutor

Fandy Aditya, Hari Ginardi, dan Abdul Munif
Departemen Informatika, Fakultas Teknologi Informasi dan Komunikasi,
Institut Teknologi Sepuluh Nopember (ITS)
e-mail: munif@if.its.ac.id

***Abstrak*—Finding Tutor merupakan aplikasi Android untuk mempertemukan Murid dengan Guru/Tutor. Penambahan fitur baru pada Aplikasi Finding Tutor bertujuan untuk mengurangi kejadian terlambat dan lupa pertemuan. Fitur yang akan ditambah adakah Schedule Request dan Event Reminder. Pada Fitur Schedule Request, memiliki tiga sub-fitur, yaitu Multiple Request yakni pencarian tutor secara terjadwal, Instant Request yakni pencarian tutor secara instan dan Manage Schedule yakni pengaturan agenda pertemuan tutor. Fitur event reminder berupa notifikasi yang mengingatkan kejadian-kejadian transaksi yang terjadi. Terdapat empat notifikasi pada fitur ini. Notifikasi adanya request tutor datang dan request tutor diterima, notifikasi pengingat pertemuan harian, notifikasi pengingat keterlambatan dan notifikasi ada request datang berdasarkan alamat yang paling sering diterima Tutor. Berdasarkan hasil pengujian aplikasi dengan metode blackbox dan kuisioner, dapat disimpulkan fitur yang dibangun berfungsi dengan baik serta kegunaan dari fitur tercapai**

***Kata Kunci*—Aplikasi Android, Notifikasi, Penjadwalan.**

I. PENDAHULUAN

FINDING Tutor merupakan aplikasi Android yang mempertemukan murid dengan Tutor/Guru. Pada aplikasi murid, sebelum mulai melakukan *request* tutor, murid harus mengisi *form* kriteria yang sesuai seperti pelajaran, tingkatan pelajaran, jam, alamat, jenis kelamin tutor dan umur tutor. Setelah selesai diinputkan, pada aplikasi Tutor, Tutor harus memeriksa ke halaman *list request* tutor yang dilakukan Murid untuk melihat pencarian apa saja yang bisa diambil oleh Tutor sesuai karakteristik dirinya. Begitu juga sebaliknya pada aplikasi Murid, Murid harus memeriksa halaman *list* transaksi untuk melihat apakah *request* Tutor sudah ada yang mengambil atau belum. Hal ini membuat Murid dan Tutor harus selalu *standy by* di aplikasi untuk mengetahui adanya *request* datang maupun *request* diterima. Terdapat aplikasi yang mirip dengan Finding Tutor, yakni Ruangguru. Ruangguru dengan salah satu fiturnya "ruangles" yang berfungsi untuk mencari guru les privat sama seperti Finding Tutor. Namun peran aplikasi hanya sebatas Murid memilih Guru les, selanjutnya akan dihubungi secara personal oleh Tim Ruangguru untuk konfirmasi maupun oleh Guru les yang bersangkutan ketika sudah mulai melakukan pertemuan.

Pada aplikasi Finding Tutor, apabila Murid ingin melakukan pencarian tutor secara terjadwal dan sekaligus, aplikasi yang sekarang belum bisa melakukannya. Oleh karena itu dibutuhkan fitur agar Murid bisa melakukan pencarian tutor secara *multiple* dengan memilih tanggal dan jam sesuai dengan yang diinginkan, serta melihat jadwal pencarian tutor yang sudah mereka buat dan mengubah apabila ada perubahan yang diinginkan. Fitur ini dinamakan Schedule Request. Dengan adanya fitur Schedule Request ini, hal yang harus diperhatikan adalah pengingat/*reminder*.

Aplikasi Finding Tutor saat ini sudah bisa melakukan pencarian tutor jauh-jauh hari sebelum hari pertemuan. Namun ketika hari pertemuan tiba, tidak ada satupun pemberitahan yang diberikan oleh sistem. Hal ini bisa menyebabkan baik Murid dan Tutor kelupaan bahwa pada hari yang bersangkutan akan diadakan pertemuan tutor. Reminder sangat penting apabila ingin melakukan sebuah penjadwalan. Reminder berupa *notification* sebagai pengingat event-event tentang pertemuan tutor. Sudah banyak aplikasi *mobile* yang menggunakan *notification*, contohnya salah satu *e-commerce* terbesar di Indonesia yakni Bukalapak. Informasi yang diberikan berupa *push notification* mengenai berupa promo produk, tips, acara, pembaruan fitur, dan beberapa informasi yang terkait transaksi. Dengan adanya *reminder*, Murid dan Tutor diharapkan akan ingat dengan pertemuan dan tidak akan telat sehingga proses belajar mengajar bisa dilakukan secara lancar.

Pada saat hari-h, *reminder* akan mengingatkan kepada Murid dan Tutor bahwa akan diadakan pertemuan pada hari itu. Reminder akan memanfaatkan GPS untuk mengetahui *real time location*, dan akan memberikan *notification* apabila Murid atau Tutor berada di lokasi yang diperhitungkan akan telat jika tidak segera berangkat. Hal ini akan mencegah terjadi kejadian lupa akan pertemuan sehingga tidak perlu terjadi tindakan *cancel* yang tentu merugikan dua belah pihak.

II. TINJAUAN PUSTAKA

A. *Finding Tutor*

Finding Tutor adalah aplikasi yang mempertemukan Murid yang ingin melakukan les privat dengan Tutor. Aplikasi ini memiliki tiga modul utama, yakni modul pencarian prioritas seleksi murid yang dibuat oleh Riska Adhita [1], pencegahan

fraud transaksi yang dibuat oleh I Nyoman Pande Wahyu Dharmawan [2], dan penentuan harga transaksi yang dibuat oleh Syah Dia Putri Mustika Sari [3].

Gambar 1. Tampilan default kalender dari Android Times Square.

B. *Push Notification*

Push Notification adalah mekanisme untuk mengirimkan informasi dari *server* ke *client* berdasarkan suatu event tertentu [4]. *Push notification* akan meningkatkan *usability* dari aplikasi *mobile*. Kehadiran *push notification* efektif untuk memberikan informasi kepada pengguna aplikasi *mobile* [5].

Seiring perkembangan jaman, setiap *smartphone* sudah banyak terinstall aplikasi, dimana pengguna harus berhadapan dengan 63,5 notifikasi setiap harinya. Oleh karena itu informasi yang akan diberikan ke pengguna harus tepat dan efisien, sehingga tidak membuat pengguna merasa teranggu oleh adanya *push notification* [6].

Sudah banyak penyedia layanan *cloud* yang menyediakan jasa untuk *push notification*. Google Cloud Messaging (GCM) yang sekarang sudah menjadi Firebase Cloud Messaging (FCM), Apple Push Notification System (APNS), Microsoft Push Notification Service (MPNS), Blackberry Push Service (BBPS), OnePush miliki Yahoo, Nokia Notifications, dan IBM MQTT [7].

Sonya R. [4] menjelaskan tentang pengembangan aplikasi penyewaan kamar dengan *push notification* via FCM. *Push notification* dilakukan ketika Web Admin mengirimkan *broadcast* ke seluruh pengguna aplikasi yang online. Walaupun informasi pada *push notification* dilakukan serentak kepada seluruh pengguna online, tidak di-*filter* berdasarkan waktu dan tempat, *push notification* tetap mendapatkan respon yang positif oleh pengguna aplikasi.

Fiona Yusina [8] memanfaatkan *push notification* untuk mengirimkan informasi terhadap suatu event pada pemain game *mobile* milik perusahaan XYZ Pte Ltd. Sistem *push notification* akan dibandingkan, yang sebelumnnya XYZ menggunakan sistem sendiri, sekarang akan menggunakan FCM untuk *platform* Android dan APNS untuk *platform* iOS. Dengan ini sistem baru terbukti bisa mengirimkan informasi hanya dengan 30 detik dimana sistem lama membutuhkan waktu 5 hari untuk mengirimkannya.

Xuan-Lam Pham [6] menggunakan *push notification* pada aplikasi "English Practice". Notifikasi yang dikirimkan berupa informasi pembaharuan aplikasi, pengingat test, pengingat review, pesan masuk baru dan komentar masuk baru. Interval pengiriman dicoba setiap 3 jam sekali, 12 jam sekali, 1 hari sekali dan 2 hari sekali. Hasil menunjukan bahwa informasi yang paling sering dibuka adalah pesan masuk baru, serta interval pengiriman 12 jam, 1 hari dan 2 hari mendapatkan hasil yang baik. Ini menunjukan bahwa informasi yang baik adalah informasi yang bersifat personal. Pengguna merasa harus membaca informasi karena benar-benar tertuju kepada dia saja. Hasil percobaan ini juga membuktikan bahwa interval pengiriman *push notification* tidak boleh terlalu sering.

C. *Firebase Cloud Messaging*

Firebase Cloud Messaging (FCM) adalah *service* yang disediakan oleh Google untuk mengirimkan pesan kepada klien secara *realtime*. Firebase Cloud Messaging sering dimanfaatkan untuk membuat aplikasi/fitur *chating* serta notifikasi. Jenis pesan FCM ada dua, yakni pesan *notification* dan pesan *data*. Pesan *notification* adalah pesan yang hanya berisi judul dan teks singkat yang berguna untuk memunculkan notifikasi pada klien. Sedangkan pesan data adalah pesan yang bisa disisipkan data didalamnya yang berupa pasangan "key", "value" sebesar maksimal 4kb. Sebelum bisa mengirim atau menerima pesan klien harus memiliki FCM Token terlebih dahulu. Google menyediakan FCM Console GUI untuk mengirim pesan langsung dari FCM *Server* kepada klien [9].

Pada aplikasi Finding Tutor akan memanfaatkan FCM API untuk mengintegrasikan kepada *server* Finding Tutor. Di bawah ini merupakan alur kerja integrasi *server* Finding Tutor dengan FCM *Server* hingga Client mendapatkan *Push Notification*.

D. *Android Times Square*

Android Times Square adalah pustaka ciptaan Square.inc yang berfokus pada Widget Kalender [10]. Pustaka ini juga memiliki penjelasan yang bagus serta forum pengguna yang masih ramai. Pembuatnya masih sangat aktif untuk menjawab pertanyaan–pertanyaan yang datang di forum. Selain itu, pustaka ini juga mudah untuk dikembangkan dikarenakan dokumentasi kode yang cukup jelas pada setiap bagian maupun fungsi–fungsi kode sumber.

Kustomisasi tampilan pada kalender ini cukup fleksibel, namun pemakaiannya cukup rumit sehingga perlu dilakukan perubahan untuk kebutuhan fungsional aplikasi yang akan dibuat.

III. DESAIN

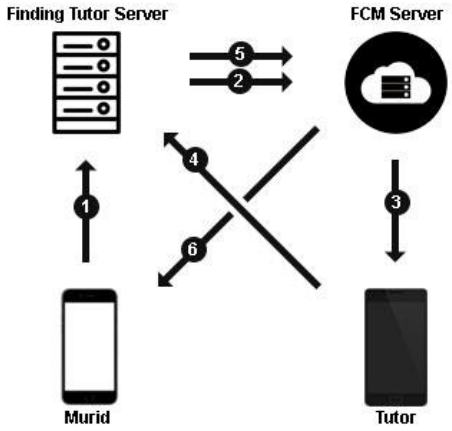
A. *Desain Schedule Request*

Fitur Schedule Request akan dibagi menjadi tiga sub-fitur, yakni Multiple Request, Instant Request dan Manage Schedule. Masing-masing sub-fitur akan memanfaatkan kalender sebagai salah satu *tools* utamanya. Kalender dibikin menggunakan pustaka *Android Times Square* untuk implementasinya. Gambar 1 menampilkan tampilan default dari kalender menggunakan pustaka Android Times Square.

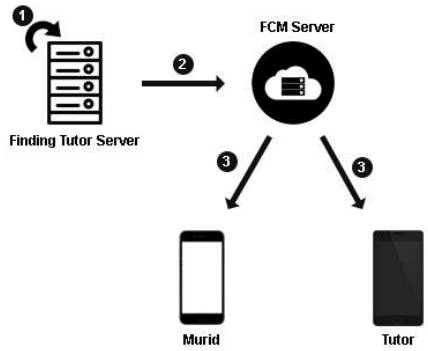
B. Desain Event Reminder

Fitur Event Reminder akan dibagi menjadi empat sub-fitur,

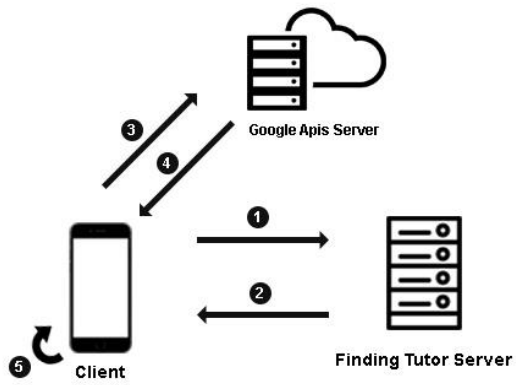
yakni notifikasi adanya *request* tutor yang datang sesuai kriteria maupun sesuai dengan alamat yang paling dikunjungi dan *request* tutor diterima, notifikasi pengingat pertemuan tutor harian dan notifikasi pengingat keterlambatan. Implementasi notifikasi akan menggunakan Firebase Cloud Messaging.



Gambar 2. Arsitektur notifikasi *request* datang dan *request* diterima.



Gambar 3. Arsitektur notifikasi pengingat pertemuan tutor di hari ini.



Gambar 4. Arsitektur notifikasi pengingat keterlambatan.

C. Multiple Request

Fitur ini membuat Murid bisa melakukan *request* tutor secara sekaligus. Murid bisa memilih jenis pemilihan tanggal satuan atau banyak. Setelah memilih tanggal akan muncul form *request* Tutor yang diisi sesuai dengan kebutuhan Murid. Murid juga bisa melakukan pembatalan tanggal dengan menekan kembali tanggal yang sudah dipilih. Semua *request* yang dilakukan akan dikirim secara sekaligus ke server dengan mengubah ke bentuk *json* terlebih dahulu.

D. Instant Request

Fitur ini membuat Murid bisa melakukan *request* Tutor saat ini juga. Dengan memilih Instant Request, akan menampilkan form *request* Tutor tanpa memilih tanggal dan waktu karena dikhususkan untuk saat ini juga. Setelah menekan tombol submit, Murid akan menunggu hingga ada Tutor yang menerima *request* dari Murid.

E. Manage Schedule

Fitur ini membuat Murid dan Tutor bisa melihat agenda pertemuan tutor yang mereka miliki. Lalu dengan menekan salah satu tanggal pada kalender akan menampilkan list dari pertemuan pada hari tersebut. Halaman detail akan muncul ketika salah satu *list* pertemuan ditekan.. Agenda pertemuan ini bisa diedit oleh Murid apabila belum ada Tutor yang menerima *request*-nya.

F. Notifikasi Request Tutor Datang dan Diterima

Notifikasi ini akan muncul ketika Murid mengirim *request* tutor dan ketika Tutor menerima *request* Murid. Arsitektur fitur ini terdapat pada Gambar 2. Berikut penjelasan dari Gambar 2.

1) Murid mengirim request Tutor ke Finding Tutor Server untuk disimpan.
2) Finding Tutor Server mengirim token Tutor yang memenuhi kriteria *request* tutor milik Murid beserta pesan bahwa ada *request* tutor masuk ke FCM Server.
3) FCM Server meneruskan pesan ke Tutor yang memiliki token yang bersangkutan
4) Tutor melakukan penerimaan *request* dan mengirim status penerimaan *request* ke Finding Tutor Server untuk diperbaharui.
5) Finding Tutor Server mengirim token Murid pemilik *request* beserta pesan bahwa *request* telah diterima ke FCM Server.
6) FCM Server meneruskan pesan ke Murid yang memiliki token yang bersangkutan.

Notifikasi *request* datang dibagi menjadi dua, yakni *request* datang sesuai dengan kriteria Tutor dan *request* datang sesuai dengan alamat paling sering diterima Tutor. Sistem akan men-*query* mengirim *push notification* ke Tutor yang sesuai alamat paling sering diterima Tutor ditentukan menggunakan batas quartil atas, yang direpresentasikan menjadi *limit* di *query*. Notifikasi *request* diterima akan ditampilkan disisi Murid ketika Tutor menerima *request* milik Murid yang bersangkutan.

G. Notifikasi Pengingat Pertemuan Tutor Hari ini

Notifikasi ini akan muncul setiap jam 06.00 ketika Murid ataupun Tutor memiliki jadwal Tutor pada hari ini. Arsitektur fitur ini terdapat pada Gambar 3 Berikut

penjelasan dari Gambar 3.

1) Finding Tutor Server melakukan pemeriksaan apakah pada hari ini disetiap *timezone* yang tersimpan ada pertemuan tutor yang harus dilakukan.
2) Jika ada, maka Finding Tutor Server mengirim token Murid dan Tutor yang akan melakukan pertemuan beserta pesan bahwa hari ini akan diadakan pertemuan pada jam dan tempat yang telah ditentukan ke FCM Server
3) FCM Server meneruskan ke Murid dan Tutor yang memiliki token yang bersangkutan

Tabel 1.
Tabel status keberhasilan pada pengujian fungsionalitas

No	Fitur	Skenario	Status keberhasilan
1	Multiple Request	1	Berhasil
		2	Berhasil
		3	Berhasil
2	Instant Request	1	Berhasil
		2	Berhasil
		3	Berhasil
		4	Berhasil
3	Manage Schedule	1	Berhasil
		2	Berhasil
		3	Berhasil
4	Notifikasi request datang dan requeset diterima	1	Berhasil
		2	Berhasil
5	Notifikasi pengingat harian	1	Berhasil
		2	Berhasil
6	Notifikasi keterlambatan	1	Berhasil
		2	Berhasil
7	Notifikasi request datang sesuai alamat tersering	1	Berhasil
		2	Berhasil

Tabel 2.
Tabel penilaian kemudahan fitur oleh pengguna

No	Fitur	Penilaian kemudahan	Penilaian pencapaian tujuan
1	Multiple Request	4.30	5.0
2	Schedule Request	4.80	5.0
3	Manage Schedule	4.37	4.6
4	Notifikasi	4.35	4.4

Sistem menggunakan *Job Scheduler* berupa *cron job* untuk mengirimkan notifikasi pada pukul 06.00 setiap harinya. *Cron job* akan menjalan *script* untuk mengecek apakah ada User yang memiliki jadwal perteman pada hari ini, jika ada maka kirim Push Notification ke User tersebut.

H. Notifikasi Pengingat Keterlambatan

Notifikasi ini akan muncul ketika Murid ataupun Tutor dikalkulasikan terlambat menemui pertemuan oleh sistem. Arsitektur dari fitur ini terdapat pada Gambar 4 Berikut penjelasan dari Gambar 4.

1) Client meminta jadwal pertemuan tutor pada hari ini ke Finding Tutor Server
2) Finding Tutor Server mengirimkan jadwal pertemuan ke Client pada hari ini
3) Client melakukan *request* ke Google Api Server untuk mengetahui waktu tempuh lokasi user dengan lokasi tujuan
4) Google Api Server mengirimkan waktu tempuh ke Client
5) Client mengkalkulasikan waktu keterlambatan dan akan menampilkan notifikasi apabila dikalkulasikan terlambat

Pada saat hari pertemuan, aplikasi akan melakukan proses secara *background* menggunakan *class Service*. Proses yang dilakukan adalah mendapatkan lokasi terkini dari User, dan melakukan *request* ke Google API Server untuk mendapatkan waktu tempuh secara interval waktu. Apabila waktu tempuh ditambah waktu sekarang melebihi waktu pertemuan, maka notifikasi akan muncul. Apabila waktu sekarang sudah melebihi waktu pertemuan, *background service* dimatikan dan menunggu hidup kembali apabila ada pertemuan di hari yang bersangkutan.

IV. UJI COBA DAN EVALUASI

Uji coba aplikasi dilakukan dengan dua cara, yakni pengujian fungsionalitas dan pengujian pengguna. Pengujian fungsionalitas menggunakan metode *blackbox*, dengan menyiapkan beberapa skenario pengujian sebagai tolak ukur keberhasilan. Hasil dari pengujian fungsionalitas dapat dilihat pada Tabel 1.

Uji coba pengguna melibatkan 10 orang dimana 5 orang bertindak sebagai Tutor dan 5 orang bertindak sebagai Murid. Setiap peran akan dipasangkan untuk mencoba aplikasi sesuai dengan skenario yang diberikan dan mengisi kuisioner penilaian setelah mencoba aplikasi. Tujuan pengujian ini adalah untuk mengetahui respon dari User serta untuk mengetahui apakah tujuan dari fitur yang dibuat sudah tercapai atau belum. Hasil dari pengujian pengguna dapat dilihat pada Tabel 2.

Berdasarkan Tabel 1 pengujian fungsionalitas di masing-masing fitur pada setiap skenario berhasil dilaksanakan. Pada hasil pengujian pengguna di Tabel 2, persentase penilaian kemudahan dari setiap fitur berurut 86%, 97%, 87.4% dan 87%. Apabila dirata-ratakan memiliki nilai 89.1%. Sedangkan untuk persentase penilaian pencapaian tujuan secara terurut yakni 100%, 100 %, 92 %, dan 88 % dengan rata-rata nilai 95%.

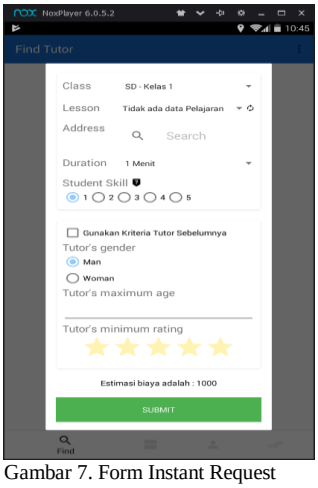
V. KESIMPULAN

Berikut kesimpulan yang didapatkan selama proses pengembangan dan uji coba:

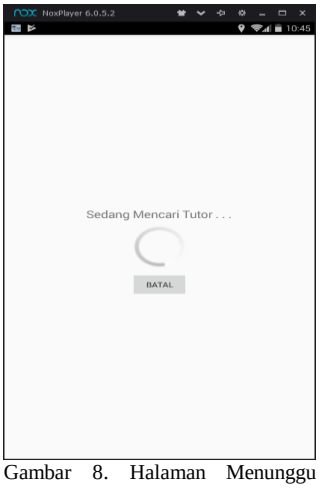
1. *Push Notification* berhasil diimplementasikan menggunakan Firebase Cloud Messaging (FCM) pada fitur-fitur notifikasi yang dibangun. Fitur notifikasi tersebut antara lain:
   a. Notifikasi ketika adanya pencarian tutor sesuai dengan kriteria Tutor, dan notifikasi ketika penerimaan pencarian tutor.
   b. Notifikasi pengingat pertemuan, dikirim menggunakan *job scheduler cron* pada *server* yang diatur setiap pukul 06.00 untuk pemanggilan *script-nya*.

Gambar 11. Antarmuka multiple request dengan mode satuan.

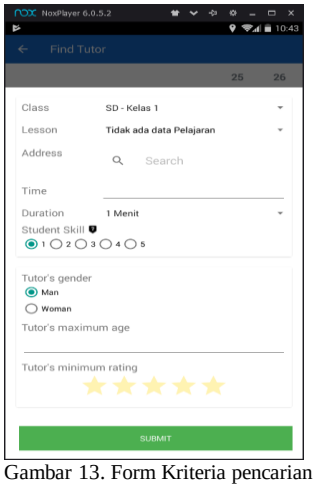
Gambar 12. Antarmuka multiple request dengan mode banyak.



Gambar 7. Form Instant Request



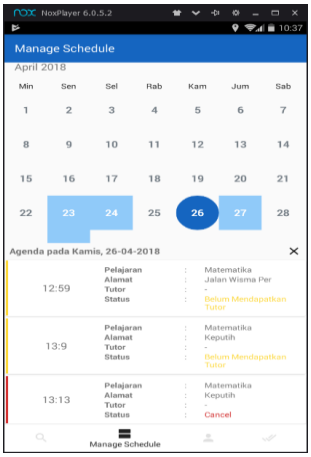
Gambar 8. Halaman Menunggu Tutor



Gambar 13. Form Kriteria pencarian dan kriteria tutor pada multiple request

Gambar 14. Membatalkan pemilihan tanggal

Gambar 9. Kalender agenda pertemuan pada halaman Manage Schedule

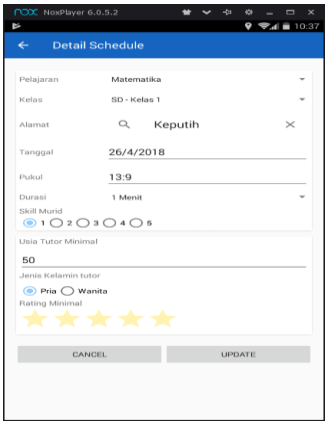


Gambar 10. List jadwal tutor pada satu tanggal

c. Notifikasi keterlambatan, membutuhkan kalkulasi untuk menentukan apakah notifikasi akan ditampilkan atau tidak. Kalkulasi dilakukan secara *background* melalui *Class Service* dengan melakukan pengambilan lokasi terkini dan memanfaatkan Google API Direction untuk mendapatkan waktu tempuh perjalanan.
d. Notifikasi pencarian tutor berdasarkan alamat yang paling sering dikunjungi, menggunakan data quartil atas dari seluruh alamat seorang Tutor sebagai batas minimal frekuensi.

2. Kalender berhasil diimplementasikan menggunakan pustaka kalender *Android Times Square*. Pustaka ini digunakan karena mendukung:
   a. Pengambilan tanggal secara banyak, digunakan untuk penjadwalan pencarian tutor
   b. Implementasi tindakan ketika interaksi menekan tanggal (*onDateClickListener*), digunakan untuk menampilkan *form* pencarian dan *list* agenda harian
   c. Penandaan tanggal (*highlight dates*), digunakan untuk menampilkan kalender agenda pertemuan

Gambar 5. Halaman Detail Schedule dengan view detail jadwal tutor



Gambar 6. Halaman Detail Schedule dengan form edit

Untuk menyempurnakan Aplikasi Finding Tutor disarankan untuk menambahkan fitur *chatting* agar Murid dan Tutor bisa langsung berkomunikasi melalui aplikasi. Selain itu disarankan juga agar halaman yang menampilkan *list* profil Tutor sesuai *filter* yang diberikan agar Murid bisa langsung memilih Tutor yang diinginkannya.

DAFTAR PUSTAKA

[1] R. Adhita, "Rancang Bangun Aplikasi Finding - Tutor Berbasis Android dan Penentuan Prioritas Seleksi Murid," Surabaya, 2017.

[2] W. Dharmawan, "Penggunaan QR Codes untuk Mencegah Fraud pada Proses Transaksi antara Penyedia Jasa Tutor dengan Konsumen Jasa Tutor pada Aplikasi Finding - Tutor Berbasis Android," Surabaya, 2017.

[3] D. Putri, "Penentuan Harga dengan Menggunakan Sistem Inferensi Fuzzy Tsukamoto Pada Rancang Bangun Aplikasi 'Finding Tutor,'" *J. Tek.*, vol. 6, no. 2, 2017.

[4] Sonya R, "Development and Evaluation of Mobile Application for Room Rental Information With Chat and Push Notification," in *ICIMTech*, 2016, pp. 7–11.

[5] Y. Chua, "An Investigation of Usability of Push Notifications on Mobile Devices for Novice and Expert Users," in *Hawai International Conference on System Sciences*, 2016, pp. 5683–5690.

[6] Xuan-Lam Pham, "Effect of push notifications on learner engagement in mobile learning app," in *IEEE International Conference on Advanced Learning Technologies*, 2016, pp. 90–94.

[7] C. Na Li, "Survey of Cloud Messaging Push Notification Service," in *International Conference on Information Science and Clound Computing Companion*, 2013, pp. 273–279.

[8] S. Fiona Yunisa, "Push Notification System to Mobile Game Player Using Distributed Event-Based System Approach," *ICSITech*, pp. 52–57, 2016.

[9] Google, "Firebase Cloud Messaging," 2017. [Online]. Available: https://firebase.google.com/docs/cloud-messaging/concept-options?hl=id. [Accessed: 21-Dec-2017].

[10] Square Inc, "Android Times Square," 2013. [Online]. Available: https://github.com/square/android-times-square. [Accessed: 18-May-2018].